PROBLEM TANITIMI

Günümüzde şehirlerdeki toplu taşıma sistemleri, vatandaşların günlük yaşamlarında önemli bir yer tutmaktadır. Ancak her şehirde farklı sistemler ve kartlar kullanıldığı için, birçok kişi için bu sistemleri kullanmak karmaşık olabilir. Bu nedenle, Türkiye genelinde geçerli bir toplu taşıma sistemi ve kartı tasarlamak ve uygulamak amacıyla bir proje planladık.

SENARYO

- İlk olarak, toplu taşıma sisteminin uygulanacağı şehirlerin bilgilerini içeren "İl" tablosu oluşturulacaktır.
- Her şehir için belediye bilgilerini tutmak üzere "Belediye" tablosu eklenecektir.
- Her şehirde kullanılan toplu taşıma araçlarını tanımlamak için "Vasıta" tablosu oluşturulacaktır.
- Vasıta türlerini belirlemek için "Vasıta Türü" tablosu eklenecektir.
- Toplu taşıma araçlarını kullanan sürücülerin bilgilerini tutmak için "Sürücü" tablosu oluşturulacaktır.
- Sürücülerin vardiyalı çalışma bilgilerini saklamak üzere "Vardiya" tablosu eklenir.
- Sürücülerin iletişim bilgilerini içeren "İletişim" tablosu eklenir.
- Yolcuların şikayetlerini kaydetmek ve takip etmek amacıyla "Şikayet" tablosu oluşturulacaktır.
- Her şehirdeki otobüs hatlarını tanımlamak için "Hat" tablosu oluşturulur.
- Mevcut ildeki tüm durakları içeren "Durak" tablosu eklenir.
- Durakların bulunduğu mahalle bilgilerini içeren "Mahalle" tablosu eklenir.
- Hareket saatleri ve tarihleri için "Hareket" tablosu oluşturulacaktır.
- Toplu taşıma araçlarını kullanan yolcuların bilgilerini saklamak için "Yolcu" tablosu eklenir
- Yolcuların kullandığı kart bilgilerini içeren "Kart" tablosu oluşturulur.
- Kart türlerini belirlemek için "Kart Türü" tablosu eklenir.

- Kartlara dolum yapabileceği dolum noktalarını içeren "Dolum Merkezi" tablosu oluşturulur.
- Yapılan dolumları kaydetmek için "Dolum" tablosu eklenir.

İŞ KURALLARI

- Bir ilde birden fazla belediye olamaz.
- Bir belediye birden fazla ilde bulunmaz.
- Bir belediyede birden fazla vasıta olabilir.
- Bir vasıta birden fazla belediyeye ait olamaz.
- Bir vasıta birden fazla türde olamaz.
- Bir türde birden fazla vasıta olabilir.
- Bir ilde birden fazla mahalle olabilir.
- Bir mahalle birden fazla ilde olamaz.
- Bir sürücü birden fazla vasıtayı kullanabilir.
- Bir vasıtayı birden fazla sürücü kullanabilir.
- Bir sürücünün birden fazla vardiyası olabilir.
- Bir vardiya birden fazla sürücüye tanımlanabilir.
- Bir sürücünün birden fazla iletişim bilgisi olmamalı.
- Bir iletişim bilgisi bir sürücüye ait olmalı.
- Bir sürücü hakkında birden fazla şikâyet yapılabilir.
- Bir şikâyet sadece bir sürücüye ait olmalı.
- Bir vasıta birden fazla hatta çalışabilir.
- Bir hatta birden fazla vasıta çalışabilir.
- Bir hatta birden fazla durak olabilir.

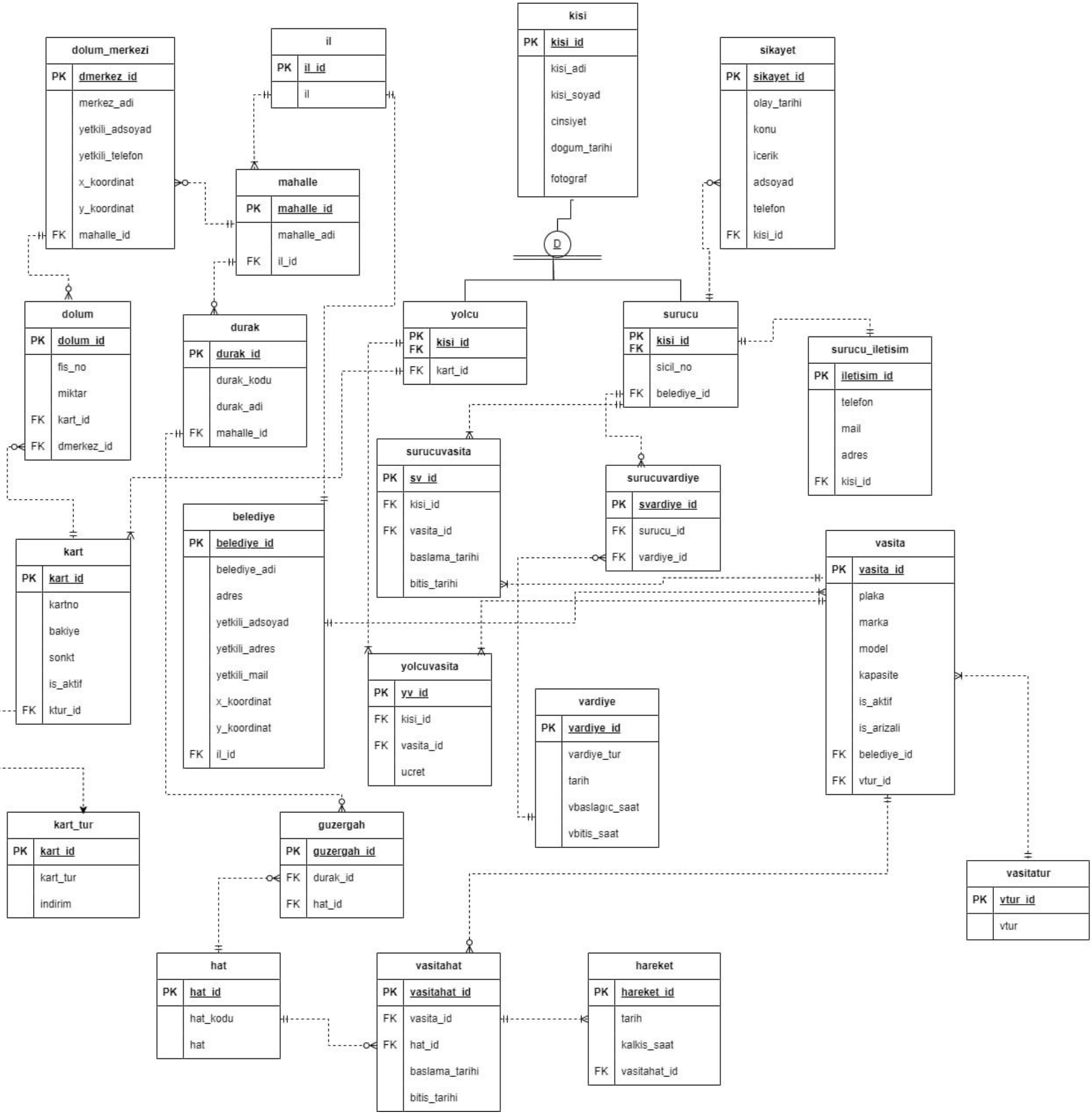
- Bir duraktan birden fazla hat geçebilir.
- Bir durak birden fazla mahallede olamaz.
- Bir mahallede birden fazla durak olabilir.
- Bir mahallede birden fazla dolum merkezi olabilir.
- Bir dolum merkezi birden fazla mahallede olamaz.
- Bir dolum merkezinde birden fazla dolum işlemi yapılabilir.
- Bir dolum işlemi sadece bir dolum merkezinde yapılabilir.
- Bir dolum işlemi sadece bir karta yapılabilir.
- Bir karta birden fazla dolum işlemi yapılabilir.
- Bir kart birden fazla kart türüne ait olamaz.
- Bir kart türünde birden fazla kart olabilir.
- Bir yolcunun bir kartı olmalı.
- Bir kart sadece bir yolcuya ait olmalı.
- Bir yolcu birden fazla vasıtaya binebilir.
- Bir vasıtada birden fazla yolcu olabilir.

İLİŞKİSEL ŞEMA

- il (il_kodu: int, il_adi: varchar)
- Belediye (belediye_id: int, belediye_adi: varchar, adres: varchar, yetkili_ad_soyad: varchar, yetkili_telefon: varchar, yetkili_mail: varchar, x_koordinat: decimal, y_koordinat: decimal)
- Vasita_Tur (vasita_tur_id: int, tur: varchar, kullanimda: boolean)
- Vasita (plaka: varchar, marka: varchar, model: varchar, yolcu_kapasitesi: int, tur_id: int, is_aktif: boolean, is_arizali: boolean)
- Kisi (kisi_id: varchar, ad_soyad: varchar, cinsiyet: varchar, dogum_tarihi: date, foto: blob)
- Surucu (kisi_id: int, sicil_no: varchar)

- Yolcu (kisi_id: varchar)
- Vardiya (vardiya_id: int, surucu_id: int, vardiya_turu: varchar, baslangic_saat: time, bitis_saat: time)
- Surucu_Iletisim (surucu_id: int, telefon: varchar, adres: varchar, mail: varchar)
- Sikayet (sikayet_id: int, tarih: date, konu: varchar, icerik: varchar, sikayet_eden_ad_soyad: varchar, sikayet_eden_mail: varchar, sikayet_eden_telefon: varchar, surucu_id: int)
- Hat (hat_kodu: varchar, hat_aciklamasi: varchar)
- Hareket (tarih: date, kalkis_saati: time, hat_kodu: varchar)
- Durak (durak_kodu: varchar, durak_adi: varchar)
- Mahalle (mahalle_adi: varchar)
- Kart (kart_numarasi: varchar, bakiye: decimal, son_kullanma_tarihi: date, tc_kimlik_no: varchar)
- Kart_Turu (kart_turu_id: int, tur: varchar, indirim_orani: decimal)
- Dolum_Merkezi (dolum_merkezi_id: int, adi: varchar, calisan_ad_soyad: varchar, telefon: varchar, x_koordinat: decimal, y_koordinat: decimal)
- Dolum (tarih: date, fis_numarasi: varchar, miktar: decimal, dolum_merkezi_id: int, kart_numarasi: varchar)

VARLIK BAĞINTI DİYAGRAMI



VERITABANINI, İÇERİSİNDEKİ VERİLERLE BİRLİKTE OLUŞTURMAYI SAĞLAYAN SQL İFADELERİ

```
create table il(
        il_id integer primary key not null,
        il varchar(5)

    )
    create table belediye(
        belediye_id integer primary key not null,
        belediye_adi varchar(140) not null,
        adres varchar(150),
        yetkili_adsoyad varchar(50),
        yetkili_telefon char(10),
        yetkili_nail varchar(25),
        x_koordinat varchar(50),
        y_koordinat varchar(50),
        il_id integer  references il(il_id)
    )
  create table kisiler(
    kisi_id integer primary key not null,
    kisi_adi varchar(25),
    kisi_soyad varchar(25),
    cinsiyet char(1) check(cinsiyet='E' or cinsiyet='K'),
    dogum_tarihi date ,
    fotograf varchar(130)
  )
```

Programın arayüzü

KULLANILAN FONKSİYONLAR

1. ekleme fonksiyonu

```
CREATE FUNCTION kisi_ekle(vid int, kisiad varchar, soyad varchar, cinsiyet char, dogum date)
RETURNS void AS $$
BEGIN
  INSERT INTO kisiler (kisi_id, kisi_adi, kisi_soyad, cinsiyet, dogum_tarihi)
  VALUES (vid, kisiad, soyad, cinsiyet, dogum);
END;
$$ LANGUAGE plpgsql;
```

Yukarıdaki fonksiyon ekleme fonksiyonudur. Kişi eklememizi sağlamaktadır.

2. Guncelleme fonksiyonu

```
create function kart_güncelleme(idk int,sonkul date)
returns void as $$
 begin
 update kart set sonkt=sonkul where kart_id=idk;
end ;
$$ language plpgsql;

create function aktifdurumu_güncelleme()
returns void as $$
begin
update kart set is_aktis=1 where sonkt>'2024.01.01';
end;
$$ language plpgsql;
```

Yukarıdaki fonksiyonlar güncelleme işlemlerinden sorumlu olan fonksiyondur.

3.Arama fonksiyonu

```
create function hat_arama(hati int)
returns table
(
    hat1_id integer,
    hat1_kodu varchar,
    hat1 varchar

) as $$
begin
return query
select * from hat where hat_id=hati;
end;
$$ language plpgsql;
```

Bu fonkisyonda hat ID sini girdiğmiz zaman bütün hat bilgilerini getirir.

```
create function kız_erkek_sayısı(sec char)
returns int as $$
declare
sayi int;
begin

 select count(*) into sayi from kisiler where cinsiyet=sec;
 return sayi;
 end;
 $$ language plpgsql;
```

Bu fonksiyon kız ya da erkek sayısını getirir.

4. Silme fonksiyonu

```
CREATE OR REPLACE FUNCTION kisi_sil(kisi INT)
RETURNS VOID AS $$
BEGIN
    IF EXISTS (SELECT 1 FROM kisiler WHERE kisi_id = kisi) THEN
        DELETE FROM kisiler WHERE kisi_id = kisi;
    ELSE
        RAISE NOTICE 'öyle bir kişi mevcut değildir.';
    END IF;
END;
$$ LANGUAGE plpgsql;
```

Bu fonksiyon silme işlemlerinden sorumlu olan fonksiyondur. Kişiler tablosundan herhangi bir kişiyi silmemizi sağlamaktadır.

KULLANILAN TRİGGER'LAR

1. Kart trigger

```
create function trigger_1()
returns trigger
as $$
begin
 insert into kart_tur (ktur_id) values (new.kart_id);
 return new;
 end;
 $$ language plpgsql;

create trigger ekleme_trigger
after insert
on kart
for each row
execute function trigger_1();
```

Yeni bir kart eklendiği zaman otomatik olarak kart türüne de eklenir.

2. Yolcu trigger

```
CREATE OR REPLACE FUNCTION trigger_2_silme_function()
RETURNS TRIGGER AS $$
BEGIN
    DELETE FROM yolcu WHERE kisi_id = OLD.kisi_id;
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_yolcu_silme
AFTER DELETE
ON yolcu
FOR EACH ROW
EXECUTE FUNCTION trigger_2_silme_function();
```

Yolcu tablosundan bir kişinin ID sini silindiği zaman bütün bilgileri otomatik olarak silinir.

3. Kart_tur trigger

```
CREATE OR REPLACE FUNCTION trigger_3_guncelleme_function()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE kart_tur
    SET ktur_id = NEW.ktur_id
    WHERE kart_tur.ktur_id = OLD.ktur_id;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_kart_guncelleme
AFTER UPDATE
ON kart
FOR EACH ROW
EXECUTE FUNCTION trigger_3_guncelleme_function();
```

Kart_tur tablosunda bir güncelleme olduğu zaman kart tablosunda da otomatik olarak güncellenir.

4. Vasita trigger

```
CREATE OR REPLACE FUNCTION trigger_4_guncelleme_function()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE vasita_tur
    SET vtur_id = NEW.vtur_id
    WHERE vasita_tur.vtur_id = OLD.vtur_id;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trigger_vasita_guncelleme
AFTER UPDATE
ON vasita
FOR EACH ROW
EXECUTE FUNCTION trigger_4_guncelleme_function();
```

Vasita_tur tablosunda bir güncelleme yapıldığı zaman otomatik olarak vasita tablosunda da güncellenir.

KULLANILAN Procedure'ler

1.procedre

```
CREATE OR REPLACE PROCEDURE BelediyeVardiyeleriListele(IN belediye_id INTEGER)
 AS
$$
 BEGIN
     SELECT * FROM vardiye WHERE belediye_id = BelediyeVardiyeleriListele.b_id;
 END;
 $$
 LANGUAGE plpgsql;
CREATE OR REPLACE PROCEDURE DolumMerkeziDolumlariListele(IN dmerkez_id INTEGER)
```

Belirli bir belediyeye ait vardiyeleri listeyen bir saklı yordam.

2.procedre

```
CREATE OR REPLACE PROCEDURE DolumMerkeziDolumlariListele(IN dmerkez_id INTEGER)
AS
$$
BEGIN
    SELECT * FROM dolum WHERE dmerkez_id = DolumMerkeziDolumlariListele.dm_id;
END;
$$
LANGUAGE plpgsql;
```

Belirli bir dolum merkezine ait dolumları listeyen bir saklı yordam.

3.procedre

```
CREATE OR REPLACE PROCEDURE HatDuraklariListele(IN hat_id INTEGER)
 AS
 $$
 BEGIN
     SELECT * FROM guzergah WHERE hat_id = HatDuraklariListele.h_id;
 END;
 $$
 LANGUAGE plpgsql;
```

Belirli bir hat üzerindeki durakları listeyen bir saklı yordam.